AMENDMENTS TO THE CLAIMS

1. (Currently Amended) A method of recovering a pointer to a non-native instruction, said method comprising:

~~executing a commit operation;~~

accessing a translation of a code sequence of non-native instructions to a code sequence of native instructions;

executing multiple commit operations during execution of said code sequence of native instructions such that there are multiple commit points associated with said translation;

registering a first address for a native instruction associated with the last occurring commit point of said commit points; ~~said commit operation; and~~

registering a second address used for recovering a non-native instruction associated with ~~said commit operation~~ the first occurring commit point of said commit points;

performing a rollback operation to said last commit point; and

restarting said translation at said first commit point.

2. (Original) The method of Claim 1 wherein said first address and said second address each comprise a program counter value for said native instruction, wherein said native instruction comprises a pointer to the effective instruction pointer value for said non-native instruction.

3. (Original) The method of Claim 1 wherein said executing and accessing further comprise:

executing a first commit operation;

accessing a first translation of non-native instructions to a first code sequence of native instructions, said first code sequence comprising a first native instruction associated with said first commit operation;

executing a second commit operation; and

accessing a second translation of non-native instructions to a second code sequence of native instructions, said second code sequence comprising a second native instruction associated with said second commit operation.

4. (Original) The method of Claim 3 wherein said second code sequence of native instructions is executed in a loop, said first code sequence of native instructions establishing conditions for said loop, wherein said second commit operation is executed each time said loop is executed.

5. (Original) The method of Claim 4 wherein said first address and said second address each comprise a program counter value for said second native instruction.

6. (Original) The method of Claim 4 wherein said second native instruction comprises a pointer to said first native instruction, said first native instruction comprising a pointer to the effective instruction pointer value for said non-native instruction.

7-10. (Canceled).

11. (Original) The method of Claim 1 wherein said native instruction comprises a first bit, a second bit, a third bit, and a plurality of pointer bits, said

first bit for causing said commit operation to be executed and a program counter value corresponding to said native instruction to be registered, said second bit for indicating whether said program counter value is to be registered, and depending on the value of said third bit, said pointer bits pointing to the effective instruction pointer for said non-native instruction or to another instruction that can be used for recovering said effective instruction pointer.

12. (Original) The method of Claim 11 further comprising:

using information in said plurality of pointer bits to identify a mode of operation associated with said translating.

13. (Original) The method of Claim 1 further comprising:

using said second address to identify a translation comprising said code sequence of native instructions.

14. (Currently Amended) A method of recovering a non-native instruction during execution of native instructions, said method comprising:

accessing a translation of a code sequence of non-native instructions to a code sequence of native instructions, said code sequence of native instructions advancing from a first occurring commit point through multiple commit points to a last occurring commit point before a rollback operation;

performing said [[a]] rollback operation to return to said last commit point using a first address to locate a native instruction associated with said last commit point; and

in conjunction with said rollback operation, using a second address to recover a non-native instruction associated with said first commit point to restart said translation.

15. (Original) The method of Claim 14 wherein said code sequence of native instructions comprises a first code sequence of native instructions and a second code sequence of native instructions, said first code sequence demarcated by a first commit operation and comprising a first native instruction associated with said first commit operation, and said second code sequence demarcated by a second commit operation and comprising a second native instruction associated with said second commit operation.

16. (Original) The method of Claim 15 wherein said second code sequence of native instructions is executed in a loop, said first code sequence of native instructions establishing conditions for said loop, wherein said second commit operation is executed each time said loop is executed.

17. (Original) The method of Claim 16 wherein said first address and said second address each comprise a program counter value for said second native instruction.

18. (Original) The method of Claim 16 wherein said rollback operation is prompted by an event that is external to said loop, said event potentially causing a change to said conditions.

19. (Original) The method of Claim 18 wherein said event is a direct memory access operation.

20-22. (Canceled).

23. (Currently Amended) A method of recovering a non-native instruction during execution of native instructions, said method comprising:

accessing a translation of a code sequence of non-native instructions to a code sequence of native instructions, said translation producing multiple commit points when executed ~~demarcated by a commit point~~;

taking an exception identified during execution of said translation;

as a result of taking said exception, reading an address in a register, said address pointing to a native instruction having an indicator bit and a plurality of pointer bits, wherein depending on the value of said indicator bit, said pointer bits point either to a single ~~an~~ effective instruction pointer for a non-native instruction associated with said multiple commit points ~~commit point~~ or to information that can be used for recovering said effective instruction pointer.

24. (Original) The method of Claim 23 wherein said information comprises another instruction that points to said effective instruction pointer.

25. (Original) The method of Claim 23 wherein said information comprises a subroutine that provides said effective instruction pointer.

26. (Original) The method of Claim 23 wherein said translation comprises a first code sequence of native instructions that establishes conditions for a second code sequence of native instructions that execute in a loop.

27. (Original) The method of Claim 26 wherein said address points to an instruction within said second code sequence, said instruction within said second code sequence having pointer bits that point to an instruction within said first code sequence, said instruction within said first code sequence having pointer bits that point to said effective instruction pointer.

28. (Original) The method of Claim 26 wherein said address points to an instruction within said second code sequence that has pointer bits that point to a subroutine that provides said effective instruction pointer.

29. (Original) The method of Claim 23 wherein said translation comprises a first code sequence of native instructions that calls a second code sequence of native instructions as a subroutine, wherein said address points to an instruction within said first code sequence that has pointer bits that point to said effective instruction pointer.